WEST

Generate Collection Print

L16: Entry 1 of 3 File: USPT Feb 5, 2002

DOCUMENT-IDENTIFIER: US 6345288 B1
TITLE: Computer-based communication system and method using metadata defining a control-structure

Brief Summary Text (23):
Another approach to automating communications and data transfers is shared replicated database systems such as Lotus Notes and Collabra Share. With these systems, information to be communicated is entered via a client program into one or more databases which may reside locally on client computers or on network server computers. These databases are then replicated to other server computers or local client computers throughout the system so that the data can be easily accessed by any other user of the system who needs the information and has the proper access privileges. Access privileges are controlled by one or more system administrators via the system servers. Some of these systems, notably Collabra Share, also allow users to "subscribe" to specific databases. These users can receive an e-mail notification from a database agent monitoring the database when a new entry or a certain condition has been made in that database. These systems may also employ electronic forms and forms processing languages to structure the data being entered into a database, and to take programmable actions based on the data entered. The architecture of these systems is designed for groups of users to share information related to specific topics, and to automate the transfer of data between different computer applications used by an organization. For this reason the core data structure of the architecture is a subject database or "forum". Each subject database covers a number of related interest topics under which all entries in the database are categorized. All copies of any subject database are synchronized throughout the system when data in any one copy has been changed.

Brief Summary Text (24):
While suitable for information sharing amongst the members of a group, this architecture is not well suited for automating communications relationships among a large number of information providers and consumers. First, all the providers and consumers need to be interconnected through the system in order to communicate. This could be done by having all providers and consumers enroll in one large system in which they all had access privileges. In such a system each provider would need to have at least one subject database for communicating with his/her consumers. This enormous number of subject databases would then need to be replicated among the large number of servers required to service the complete population of the system, which would quickly overwhelm the capacity of the servers or network to handle replication. A more realistic alternative would be to have each provider or group of providers operate and administer their own system, making their internal subject databases available to consumers via public data networks such as the Internet. Consumers would use the system client software to "subscribe" to the subject databases of each provider with which they desire to communicate. Only the subject databases a consumer subscribed to would be replicated on his/her desktop. This solution would spread the replication load to a large number of servers, each handling a smaller amount of traffic. However, each server would now have to manage replication for a large number of external consumers as well as internal group members. There is no easy way to distribute this replication load to the consumer's computer. Second, subject databases do not allow the consumer to control and filter the incoming communications from providers. Consumers must still scan the databases for items of interest. Providers could overcome this by creating a subject database for each interest topic, but the additional administrative and server replication overhead would strongly discourage this. Third, because notification of new information is handled via a separate application, e-mail, the consumer is forced to coordinate notification and data storage/response among two communications systems. Fourth, since subject databases are replicated from the servers, they do not give consumers an easy way to copy or transfer them to other consumers. Finally, because the entire system depends on server-based replication, administrative changes or

reconfigurations of these servers such as system name or address changes require administrative updates to all subscribing consumers, a job which consumers must handle manually.

Detailed Description Text (3):
There is illustrated in FIG. 1 a first embodiment of a system of the present invention which automatically updates a database in a consumer computer with 10 information from a provider computer. Numerous providers and consumers exist in the system of the present invention. However, since all communications can be separated into transfers between a single provider and consumer, the design and operation of the system is illustrated with only one provider and one consumer, except as otherwise noted. As illustrated in FIG. 1, a provider computer 1 includes a provider database 11 of communications control information which it desires to disseminate or make accessible to one or more consumers. A consumer computer 2 includes a consumer database 21 of communications control information received from providers and stored by the consumer. The organization, structure, and content of the provider database 11 and consumer database 21 are discussed below. The provider computer 1 is connected through a communications network 3 to the consumer computer 2. Any communications network 3 may be used to connect the provider computer 1 and the consumer computer 2, including direct network connections, server-based environments, telephone networks, the Internet, intranets, local area networks (LANS), wide area networks (WANS), the World Wide Web, other webs, and even transfers of data on physical media such as disks or computer-readable paper outputs via postal communications networks. The particulars of the communications network illustrated as preferred embodiments are not limiting features of the invention. However, the Internet and World Wide Web provide existing capabilities between computers sufficient to provide the necessary connections. For this reason, the description of the present invention is based on this communications medium, which should be understood to be used for purpose of illustration only. Organization and operation of the Internet and communications over the Internet are discussed generally in Kris Jamsa and Ken Cope, Internet Programming (1995) and Marshall T. Rose, The Internet Message: Closing the Book with Electronic Mail (1993), which are incorporated herein by reference. Communications over the World Wide Web are discussed generally in John December and Neil Randall, The World Wide Web Unleashed (1996), which is incorporated herein by reference. Additionally, the illustrated embodiment is not limited to the specific networks known as the "Internet" and "World Wide Web", but relate to internet, intranet and web networks generally. A specific feature of this invention is that it is easily adaptable to control and automate communications via any type of communications network. In addition, it can select a preferred communications network and message encoding format to be used for a specific communications transaction, as further described below.

Detailed Description Text (8):
Appropriate programs executing on the provider computer 1 and the consumer computer 2 perform the functions necessary to transfer, maintain, and update the information at both locations. A program represents a set of stored instructions which are executed in a processor of the computer to process data, transmit and receive data, and produce displays. The provider program 12 operates to transmit changes in information stored in the provider database 11 at the provider computer 1. When changes are made to the information and the database, the provider program 12 operates to disseminate the changed information through the communications network 3. In the pushing method, the provider program 12 transmits the changed information, for example through e-mail, to the consumer computers 2 of all intended recipients. In the pulling method, the changed information is stored on a distribution server 32, such as a web server, which then can be accessed by the consumer computer 2. Any type of distribution server may be used, including network file servers, FTP servers, gopher servers, and so on. The type of distribution server used is not a limiting feature of the invention. The consumer program 22 will typically poll the distribution server 32 to determine whether the information has changed. This polling operation can be as simple as issuing a Web server HTTP file date request and comparing this with the file date of the last update. Polling is controlled by the information transferred from the provider program to the consumer program as further described below. Upon receipt of changed information, the consumer program 22 operates to perform certain functions with regard to that changed information. Principally, the information is stored in consumer database 21 on the consumer computer 2 for future reference and usage in controlling and automating communications between the consumer and provider. Furthermore, the information may be presented to a user at the consumer location, so that the user will be notified of the changed information. The information can be presented in a number of manners, including display or printing by the consumer program, sending an e-mail or voice-mail message to the user, paging the user, and other notification methods.

Detailed Description Text (65):
The second alternative is to use a centralized server or set of servers to produce unique system IDs as required. This approach has the advantage of creating a centralized registry of unique system IDs. Such a registry has other applications within a communications object system, as further described below. The architecture for centralized system ID assignment is illustrated in FIG. 5. A central system ID server 42 contains a database of system ID assignments 41. The system ID database 41 could be replicated across a group of ID servers 42 at various nodes of a communications network 3 to improve performance as the number of users increases. Upon initial installation, each provider program 12 or consumer program 22 sends a request 44 via the communications network 3 to the ID server 42 for a unique system ID 43. The ID server 42 returns a response 45 to the requesting program. The requesting program then saves the system ID in the provider database 11 or consumer database 21. This system ID 43 is shown in FIG. 3 as the SystemID attribute of the Database class 100. Within the database, the provider and consumer programs 12, 22 can include a function for assigning a separate unique system ID value to each instance of a communications object 110 or any class that will become a component of a communications object. In FIG. 3, these classes include the rule 140, method 141, page 142, element 143, and typeDefinition 144 classes. Again, this function is the equivalent of an automatically-generated unique key field ID in a conventional database management system. Since the provider's system ID 100 is unique for the entire communications system, and since each instance of a communications object system ID 110 or any or any component class system ID is unique within the provider's database, the combination of these system IDs creates a hierarchical indentification system capable of uniquely identifying every communications object instance or object component class instance throughout the communications system. This unique ID for any communications object or communications object component will be referred to as its UID.

Detailed Description Text (163):
Just as with a standard communications object, when the external components or component objects of a synthesized object change after the receipt of an updated object from the external provider, those changes trigger the update association rule, described above. Thus the changes are propagated upward to the components and communications objects which contain them using the update association method (FIG. 10B). In this fashion synthesized communications objects transmit the changes to their external components in the same fashion as they do with their internal components.

Detailed Description Text (241):
When the provider wishes to transmit information related to a notification element, the provider uses the edit selected element form (342, FIG. 9) to create or edit the message elements 211 or other elements associated with one or more notification element. The provider does this for all messages or other notification events the provider wishes to transmit in a particular communications object distribution. Of course any other communications object or object component changes will also be transmitted in the same distribution operation.

Detailed Description Text (274):
Referring to FIG. 3, the preceeding discussion of data exchange has focused primarily on data exchange events initiated either manually by the consumer or automatically by scheduled events 117. Either one of these techniques can be employed to either pull new data to the consumer or push data from the consumer to the provider. However data exchange events can also be triggered automatically by data exchange rules 140. Perhaps the most common example is the update association rule discussed above. The update association rule (FIG. 10B) ensures that changes made within the provider database 11 are distributed to all associated recipients 120 via all associated communications objects 110. When the provider and consumer programs 12, 22 and databases 11, 21 are combined, a corresponding data exchange rule 140 can be employed by a consumer communications object 110 to monitor changes to one or more provider elements 143. Such a data exchange rule creates an association between an element 143, a communications object 110, and a data exchange method 141. When the version value of the element 143 changes, the rule 140 is invoked which calls the data exchange method 141 of the communications object 110. The data exchange method 141 can then produce a message object 110 to transmit the changed data back to the provider program 12. Alternatively, a message object or another type of structured message can be transmitted via any available communications network to any other address the provider designates. In this fashion a communications object consumer who is also a provider can automatically update all of his/her communications object relationships associated with a particular element 143 just by editing the element 143. A common universal example is

change-of-address notifications. With any other communications object network, such as postal systems, telephone systems, fax systems, or e-mail systems, a change of address involves significant human labor on the part of the party whose address is changing to notify all his/her communications partners. For their part, each communications partner must update their own records when a change-of-address notification is received. When data exchange rules 140 are employed in the present invention, every communications partner is updated automatically whenever relevant communications data elements change. This includes both partners who are recipients 120 of the user's own communications objects 110, as well as partners from whom the user has received a communications object 110 employing data exchange rules. As the number of users of a communications object system grows, the overall human labor savings involved in this automatic two-way updating of contact data can become very significant.

Detailed Description Text (277):
Because data exchange control is one of the core functions of the present invention, it can be applied in many unique ways. An example is offline viewing of World Wide Web content. Existing software programs such as Freeloader from Individual Inc. and WebEx from Travelling Software Corporation allow Internet users to download to their local hard disk selected Web pages together with all or a selected subset of the helper files (graphic files, sound files, multimedia files, etc.) used on that page. These programs are called "offline browsers" because they allow the reader to view all or a majority of the web page's content while offline. This speeds up viewing times and reduces online access charges. These programs can also monitor the web page using a polling interval set by the user and download new versions when the page is updated.

Detailed Description Text (278):
This functionality can be significantly improved upon using a communications object system and data exchange controls. First, in existing offline browsers, a user must select specific web pages for monitoring and downloading, and there is no selective notification about changes to those pages. With a communications object system, a single communications object 110 can allow the consumer to select from a variety of notification elements (201, FIG. 4), each with its own notification control. Second, most existing offline browsers require the user to choose the "depth" of linked pages and helper files that will be downloaded for new or changed pages. This is strictly a guessing game for users, who have no clear idea what additional pages or supporting helper files they may be interested in. With a communications object system, the provider of the content can create a data exchange method 141 that presents the relevant downloading options to the consumer, and then intelligently sets a polling interval and selects the content the provider knows is related to the consumers expressed interest. Third, with offline browsers, downloaded web pages have no feedback component other than hyperlinks contained on these pages. With a communications object system, the downloaded data can include or be linked to communications objects 110 and their components. Thus feedback can happen both manually using data exchange elements 143 that return message objects 110, and automatically using reporting control. Reporting control is further discussed below.

Detailed Description Text (292):
When the provider program 12 and consumer program 22 are combined, a communications object system can also automate the exchange of communications objects 110 between two providers. This is a common need analogous to the exchange of business cards between individuals. Although only one business card need be exchanged to create a communications relationship, a two-way exchange provides the greatest number of communications options in both directions. This synchronization can be accomplished by the use of a special method 141 called an autoexchange method. An autoexchange method 141 operates as both a receipt method 141 and a data exchange method 141. The the data structures involved are illustrated in FIG. 3. When a provider program 12 first receives a communications object 110 with an autoexchange method 141, it first compares the database system ID of the communications object 110 with those of its recipient instances 120. If the database system ID is already present, the relationship is already established, and the autoexchange method is ignored. If not, the autoexchange method 141 is executed according to the receiving provider's preferences. Such preferences can be specified using the provider preferences form (316, FIG. 9) and stored in the global preferences instance 103. One preference may be to automatically generate and transmit a default communications object 110 instance back to the first provider. Another preference may be to generate a message element (211, FIG. 4) for use with the notification report (630, FIG. 13) to notify the receiving provider of the autoexchange request. The headline of the message element (211, FIG. 4) could be linked to the create new recipient form (311, FIG. 9). This form would serve as the input form for the autoexchange method 141. The received communications object 110 could supply

the attributes necessary for the recipient instance 120 on this form. Therefore the receiving provider need only select the preferences for the communications object or objects 110 to be returned to the first provider. When this form is submitted, the autoexchange method will trigger the distribution of this communications object as described above.

Detailed Description Text (321):
In general, providers only need to control archiving in the provider database 11 and consumers only need to control archiving in the consumer database 21. However, archiving control can also be combined with distribution control and data exchange control as a way to ensure the versions of a communications object in a provider database 11 and a consumer database 21 stay synchronized. This is another aspect of version monitoring, discussed above. Version monitoring is desirable because it is possible for the consumer to miss versions of a communications object instance 110. For example, if the communications object instance 110 is distributed via the push technique, communications network errors may cause the transmission to fail. If the communications object instance 110 is distributed via the pull technique, communications network errors or distribution server errors may also prevent an update from occuring, although the polling retry process can frequently correct this. A more likely scenario is that either the consumer computer 2 or the consumer program 22 is not operated by the consumer during one complete version update cycle by the provider. For example, if the provider updates the communications object once per day, but the consumer does not operate the consumer program 22 for a week, the consumer may have missed six communications object versions. Finally, a communications object version could become corrupted in the consumer database 21.

Detailed Description Text (322):
If a uniform version value algorithm is used throughout the communications object system to increment the value of version attributes of communications object instances 110 or its components, recovery of lost or missing versions is straightforward. When the push technique of updating is used, recovery is accomplished using a version monitoring rule 140 and version monitoring method 141 at the consumer program 22. These can be implemented by the provider or the consumer. The version monitoring method 141 operates as a specialized data exchange method 141, explained above. The version monitoring rule 140 is associated with the communications object 110 so it is triggered each time an update is received. The version monitoring rule 140 executes the version monitoring method 141 which compares the version value of the update received with version value of the most recent communications object 110 stored in the consumer database 21. If the version value algorithm indicates that a version value is missing in the sequence, the version monitoring method 141 generates a message object 110 containing a data exchange element 143 specifying the missing version values and the system ID of the consumer. The version monitoring method 141 then transmits the message object back to the provider program 12. Here, the message object executes a version monitoring receipt method 141. The version monitoring method 141 then executes the communications object instance generation and transmission routine (FIG. 12) for the specified missing version communications object versions and the specified recipient 120. When these new instances are received by the consumer program 22, synchronization is restored.

Detailed Description Text (323):
When the pull technique of updating is used, the steps are slightly different. In this case the version monitoring method 141 at the consumer program 22 needs to be able to determine the network address of the missing communications object versions on a distribution server 32. This can be done in several ways. A first technique is to use a version naming algorithm to derive the address from a combination of the network address of the current communications object 110 and the missing version value. For example, if the network address of the current communications object instance 110 is http://company.com/commobject.cos and the missing version value is 3481, then the computed network address would be http://company.com/commobject3481.cos. This requires two minor modifications to the communications object instance generation and transmission routine (FIG. 12). First, a step is added in which the version naming algorithm is used to rename a copy of the previous communications object version instance 110 stored on the local or network file system. Alternatively, the routine could regenerate this previous instance if it were missing. Second, at the conclusion of the routine, both the current and renamed previous version of the communications object 110 are transferred to the distribution server 32. Now when a new communications object instance 110 is received at a consumer program 22, the version monitoring rule 140 executes a version monitoring method 141. The version monitoring method 141 first determines if any versions are missing. If so, it uses the version naming alrogrithm to

determine the network addresses of each missing communications object version. It then executes a polling operation for each missing communications object from the distribution server 32, restoring synchronization.

Detailed Description Text (324):
A second technique is to store the date and network address of previous versions in the communications object 110 itself. This approach has the advantage of allowing any number of version naming algorithms to be used. This can be done with three minor modifications to the communications object instance generation and transmission routine (FIG. 12). The first modification provides that each time a new version of a communications object instance 110 is generated, the version naming algorithm is used to generate a unique network address name for this version. This unique network address name together with the version value and current date and time is stored as a archive composite type element 143 contained in the communications object 110. This element is itself maintained by an archive rule 140 such that only n instances, or only instances newer than X interval, are stored. The second modification is the same as described above, i.e., the communications object instance generation and transmission routine uses the unique network address name to rename a copy of the previous communications object version instance 35 stored on the local or network file system. Alternatively it could regenerate the previous instance of the communications object instance 35 it were missing. Finally, at the conclusion of the routine, both the current and renamed previous version of the communications object instances 35 are be transferred to the distribution server 32. Now when the new communications object instance 35 is received at a consumer program 22, the version monitoring rule 140 executes a version monitoring method 141 as above. The version monitoring method 141 compares the version value attributes of the archive elements 143 contained in the updated communications object instance 35 with those in the consumer database 21. If any are missing, the version monitoring method 141 uses the network address stored in the archive element 143 to execute a polling operation for each missing communications object from the distribution server 32, restoring synchronization.

Detailed Description Text (337):
Service objects can wholly contain the services they offer, or they can represent the services of one or more servers available in the communications object system. In the latter case, the service object forms the basis for communicating the necessary information to the server so that the service can be provided. The latter case is also particularly useful because the service object can abstract or "encapsulate" the interface to the server or servers, removing the need for either the provider or consumer to deal with this complexity. A service object may represent services provided by a network of related servers, for example a replicated directory database such as the Internet's Domain Name Service (DNS). The service object can then also abstract and automate the process of choosing one of the network servers as a current partner server which will result in optimal performance and minimal network traffic. Referring to FIG. 3, this feature is accomplished by including a receipt method 141 in the service object (815, FIG. 17) having one or more algorithms for determining the optimal partner server. Such algorithms can access elements 143 in the provider database 11 or consumer database 21 for necessary input data such as the user's geographic location, time zone, network address, and so on, or they can prompt the user for such data via input forms. The service object's receipt method may also use a data exchange method 141 to query a reference server, perform network packet timing tests, or use other techniques to determine the optimal partner server. The same approach can also be used to determine one or more backup partner servers to use in case the primary partner server is unavailable. The particular algorithm or method for determing an optimal partner server or backup partner servers is not a limiting feature of the invention. Once determined, the receipt method can save this data in the provider database 11 or consumer database 21 as one or more element preference instances 147. These element preference instances can then be accessed by the service object's update method 141 or any of its other methods whenever it needs to call operations at the partner server.

Current US Cross Reference Classification (1):
707/1

Current US Cross Reference Classification (2):
707/10

Current US Cross Reference Classification (3):
707/102

Current US Cross Reference Classification (4):

707/104.1